

## 9. Pamela

**Program Name: Pamela.java**

**Input File: pamela.dat**

Pamela has recently started learning about 2-dimension arrays that contain rows and columns of data. Before she starts with more complex operations, she wants to practice with the basics. Her goal is to be able to consistently compute the sums of individual rows and columns and find minimum values for the columns and maximum values for the rows for arrays of various sizes. The following 5 x 4 sample shows the results she will calculate:

Columns ↘ Rows ↓	0	1	2	3	Row Sums	Row Mins
0	160	918	572	587	2237	160
1	817	155	703	903	2578	155
2	471	468	962	311	2212	311
3	890	575	532	128	2125	128
4	266	259	442	167	1134	167
Col Sums	2604	2375	3211	2096		
Col Maxs	890	918	962	903		

Can you produce the same result so Pamela can check hers?

**Input:** The first line is a positive integer  $1 \leq T \leq 10$ , the number of test cases in the data file. That will then be followed by T sets of data. For each dataset, the first line will contain 2 integers: the number of rows (R) and the number of columns (C) with  $2 \leq R, C \leq 15$ . The dataset continues with R rows, each containing C integers (N) to populate the individual array cells with  $0 \leq N \leq 1000$ .

**Output:** For each test case, the first line contains a case number, formatted as shown in sample. The next four lines contain the row sums, the row minimums, the column sums, and the column maximums. The next two lines contain the overall minimum and the overall maximum. All lines must be labeled and formatted as shown below with integers displayed in right-aligned fields that are 7 positions wide and no additional spacing. The final line for each test case will contain 20 equal signs "===== ". There are no blank lines.

**Sample input:**

```

2
5 4
160  918  572  587
817  155  703  903
471  468  962  311
890  575  532  128
266  259  442  167
7 9
286  523  961  240  866  234  252  688  437
922  182  702  925  651  613  820  477  580
10  516  533  639  239  51  538  300  268
620  473  663  705  10  210  85  597  613
459  608  828  465  669  327  932  174  950
984  413  465  788  958  760  817  402  531
571  511  757  62  581  444  650  271  65

```

*See next page for sample output...*

## UIL – Computer Science Programming Packet – Invitational B - 2022

*Pamela continued...*

### Sample output:

Case: 1

Row Sums: 2237 2578 2212 2125 1134

Row Mins: 160 155 311 128 167

Col Sums: 2604 2375 3211 2096

Col Maxs: 890 918 962 903

Min Mins: 128

Max Maxs: 962

=====

Case: 2

Row Sums: 4487 5872 3094 3976 5412 6118 3912

Row Mins: 234 182 10 10 174 402 62

Col Sums: 3852 3226 4909 3824 3974 2639 4094 2909 3444

Col Maxs: 984 608 961 925 958 760 932 688 950

Min Mins: 10

Max Maxs: 984

=====

## 6. Rhea

**Program Name:** Rhea.java

**Input File:** rhea.dat

Rhea is at the mall, where the floor tiles are many different colors. As a fun game, Rhea only wants to walk on tiles in order of the rainbow (Red, Orange, Yellow, Green, Blue, Indigo, Violet). If Rhea is on a certain color tile, she is only willing to move to an adjacent tile if it is the same color, or the next color in rainbow order (This order is not cyclical). Given this restriction, help Rhea determine if such a path exists from her current location to the smoothie shop.

**Input:**

The first line will consist of a single integer  $T$  ( $1 \leq T \leq 10$ ), the number of test cases to follow.

Each test case begins with two integers  $L$  and  $W$  ( $1 \leq L, W \leq 50$ ) on their own lines, denoting the length and width of the mall in floor tiles.

The next  $L$  lines will each consist of a row with  $W$  floor tiles.

Rhea begins on a red tile, and her location will be denoted by a '\*'.

The smoothie shop is on a purple tile, and its location will be denoted by a '#'.

Any other tiles will be represented with the first letter of their respective colors.

**Output:**

Output `yes` if it is possible for Rhea to reach the smoothie shop from her current location, and `no` otherwise.

**Sample input:**

```
3
5 5
*RRRR
VVVVO
YYYYY
GGGGG
BIVV#
2 5
*YGG#
OYVBI
1 5
*YGB#
```

**Sample output:**

```
yes
yes
no
```